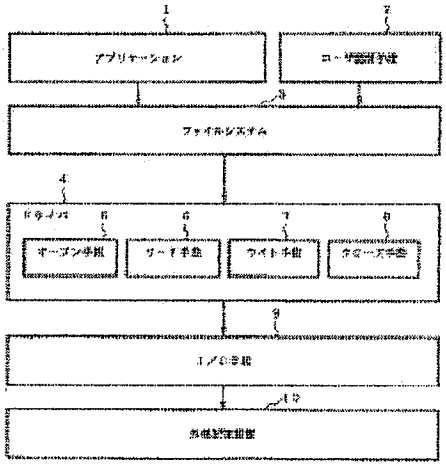
FILE ACCESS SYSTEM AND RECORDING MEDIUM

Publication number: JP10301856 (A)
Publication date: 1998-11-13
Inventor(s): SASAKI TAKAOKI; YAMANAKA YUSUKE +
Applicant(s): FUJITSU LTD +
Classification:
- international: ***G06F12/00; G06F12/14; G06F21/24;*** **G06F12/00; G06F12/14; G06F21/00;** (IPC1-7): G06F12/00; G06F12/14
- European:
Application number: JP19980045268 19980226
Priority number(s): JP19980045268 19980226; JP19970046724 19970228

Abstract of **JP 10301856 (A)**
PROBLEM TO BE SOLVED: To easily and partially access a deciphered file at a high speed and to raise a security level by taking out a part with an access request from a ciphered file by a read means or a write means, deciphering it or deciphering and writing it, then ciphering it and writing it back. SOLUTION: When data read from a pertinent file in an external storage device 10 by an I/O means 9 corresponding to a read request from an application 1 are ciphered, the read means 6 returns the deciphered data of the same size to the application 1 of a read request origin or the like. Also, when the data read from the pertinent file in the external storage device 10 by the I/O means 9 corresponding to a write request from the application 1 are ciphered, the write means performs subscribing to the deciphered data of the same size and writes them back to the pertinent file in the external storage device 10.



(19)日本国特許庁（ＪＰ）　(12)**公開特許公報**（Ａ）　(11)特許出願公開番号

特開平10－301856

(43)公開日　平成10年(1998)11月13日

(51)Int.Cl.[6]		識別記号	FI		
Ｇ０６Ｆ	12/14	３２０	Ｇ０６Ｆ	12/14	３２０Ｂ
	12/00	５３７		12/00	５３７Ｈ

審査請求　未請求　請求項の数６　ＯＬ　（全 13 頁）

(21)出願番号　特願平10－45268

(22)出願日　平成10年(1998)２月26日

(31)優先権主張番号　特願平9－46724

(32)優先日　平９(1997)２月28日

(33)優先権主張国　日本（ＪＰ）

(71)出願人　000005223
富士通株式会社
神奈川県川崎市中原区上小田中４丁目１番１号

(72)発明者　佐々木　孝興
東京都港区芝浦四丁目15番33号　株式会社富士通ビー・エス・シー内

(72)発明者　山中　祐介
東京都港区芝浦四丁目15番33号　株式会社富士通ビー・エス・シー内

(74)代理人　弁理士　岡田　守弘

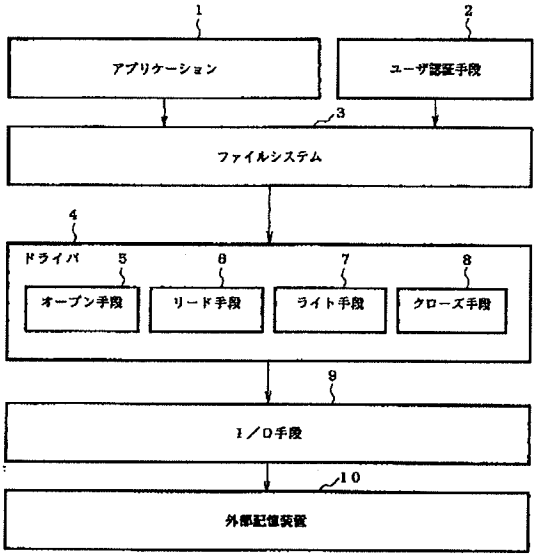
(54)【発明の名称】　ファイルアクセスシステムおよび記録媒体

(57)【要約】

【課題】　本発明は、オープン命令が発行されたファイルの一部のデータに対するアプリケーションから発行されたリード／ライト命令に対してアクセスするファイルアクセスシステムおよび記録媒体に関し、アクセス依頼に対応してリード手段あるいはライト手段が暗号化されているファイルからアクセス依頼のあった部分を取り出して復号化あるいは復号化してライトした後に暗号化して書き戻し、暗号化されたファイルへのアクセスを高速かつ簡易にしかも部分的に行い、セキュリティレベルを高めることを目的とする。

【解決手段】　アプリケーションから発行される前記リード命令に対して、リード処理対象のファイルが暗号化対象かを判定し、暗号化対象である場合には、暗号化されているファイルのうち、リード命令の処理対象のデータを復号化して、リード命令の発行元に返すリード手段を備えたるように構成する。

本発明のシステムブロック図



【特許請求の範囲】
【請求項１】オープン命令が発行されたファイルの一部のデータに対するアプリケーションから発行されたリード命令に対して、実際のファイル内の指定されたデータをリード命令を発行したアプリケーションに返す、ファイルアクセスシステムにおいて、
アプリケーションから発行される前記リード命令に対して、リード処理対象のファイルが暗号化対象かを判定し、暗号化対象である場合には、暗号化されているファイルのうち、リード命令の処理対象のデータを復号化して、リード命令の発行元に返すリード手段を備えたことを特徴とするファイルアクセスシステム。
【請求項２】オープン命令が発行されたファイルの一部のデータ領域に対する、アプリケーションから発行されるデータのライト命令に対して、処理対象のファイルの一部のデータ領域に対して、指定されたデータを書き込むファイルアクセスシステムにおいて、
アプリケーションから発行される前記ライト命令に対して、ライト命令の処理対象のファイルが暗号化対象か判定し、暗号化対象である場合には、暗号化されているファイルのうち、ライト命令の処理対象のファイルの一部のデータ領域を含み、かつ、暗号化の単位の長さのデータを読み出して復号化し、ライト命令で指定されたデータを該復号化したデータに上書きし、暗号化して、処理対象のファイルのデータ領域に格納するライト手段を備えたことを特徴とするファイルアクセスシステム。
【請求項３】前記ファイルから読み出した暗号化されていたデータを復号化して同一サイズのデータにし、あるいは暗号化されていないデータを暗号化して同一サイズのデータにしてファイルに書き込むことを特徴とする請求項１あるいは請求項２記載のファイルアクセスシステム。
【請求項４】ユーザ毎あるいは前記ファイル毎に前記リード手段による復号化あるいは前記ライト手段による復号化／暗号化の可あるいは不可を登録したことを特徴とする請求項１ないし請求項３記載のいずれかのファイルアクセスシステム。
【請求項５】コンピュータを動作させて、
オープン命令が発行されたファイルの一部のデータに対するアプリケーションから発行されるリード命令に対して、リード処理対象のファイルが暗号化対象かを判定し、暗号化対象である場合には、暗号化されているファイルのうち、リード命令の処理対象のデータを復号化して、リード命令の発行元に返すリード手段を機能させるプログラムを格納した記憶媒体。
【請求項６】コンピュータを動作させて、
オープン命令が発行されたファイルの一部のデータに対するアプリケーションから発行されるデータのライト命令に対して、ライト命令の処理対象のファイルが暗号化対象かを判定し、暗号化対象である場合には、暗号化されているファイルのうち、ライト命令の処理対象のファイルの一部のデータ領域を含み、かつ、暗号化の単位の長さのデータを読み出して復号化し、ライト命令で指定されたデータを該復号化したデータに上書きし、暗号化して、処理対象のファイルのデータ領域に格納するライト手段を機能させるプログラムを格納した記憶媒体。
【発明の詳細な説明】
【０００１】
【発明の属する技術分野】本発明は、ファイルにアクセスするファイルアクセスシステムおよび記録媒体に関するものである。
【０００２】
【従来の技術】従来、アプリケーションなどからアクセス依頼のあったデータについて、外部記憶装置に格納されているファイルをアクセスしている。この際、ファイルが暗号化されている場合には、ファイル内のデータを作業ファイルに一旦取り出して復号化した後、当該作業ファイルをアクセスしてアクセス処理終了後に暗号化して外部記憶装置のファイル内に書き戻すようにしていた。
【０００３】
【発明が解決しようとする課題】上述したように、従来は、外部記憶装置中のファイルが暗号化されていた場合には、当該ファイル内の暗号化されているデータを一旦作業ファイルに取り出して復号化した後、当該作業ファイルにアクセスしてアクセス処理を行い、一連の処理が終了した後、暗号化して元のファイルに書き戻すようにしていたため、アクセス依頼に対応してファイルが暗号化されていると作業ファイル作成や復号化して作業ファイルに格納などの処理が必要となり、迅速にファイルにアクセスできないという問題があった。また、復号化されたデータが作業ファイルに存在しセキュリティに欠けるという問題もあった。
【０００４】本発明は、これらの問題を解決するため、アクセス依頼に対応してリード手段あるいはライト手段が暗号化されているファイルからアクセス依頼のあった部分を取り出して復号化あるいは復号化してライトした後に暗号化して書き戻し、暗号化されたファイルへのアクセスを高速かつ簡易にしかも部分的に行い、セキュリティレベルを高めることを目的としている。
【０００５】
【課題を解決するための手段】図１を参照して課題を解決するための手段を説明する。図１において、アプリケーション１は、アクセス要求を行うものである。このアプリケーションは既存のアプリケーションであり、このシステムのためにソースコードの変更を行う必要がない。
【０００６】ファイルシステム３は、アプリケーション１からのアクセス要求に対応してドライバ４に指示してアクセス処理を行うものである。ドライバ４は、ファイ

ルの各種処理を行うものであって、ここでは、ファイルのリード処理を行うリード手段６、ファイルへのデータのライト処理を行うライト手段７などから構成されるものである。

【０００７】Ｉ／Ｏ手段９は、外部記憶装置１０のファイルからデータを読み出したり、書き戻したりなどするものである。次に、動作を説明する。

【０００８】リード手段６がアプリケーション１からのリード要求に対応して、リード要求のあったデータについて、Ｉ／Ｏ手段９によって外部記憶装置１０中の該当ファイルから読み出させたデータが暗号化されていたときは復号化したデータをリード要求元のアプリケーション１などに返し、一方、暗号化されていなかったときはそのままのデータをリード要求元のアプリケーション１などに返すようにしている。

【０００９】また、ライト手段７がアプリケーション１からのライト要求に対応して、ライト要求のあったデータについて、Ｉ／Ｏ手段９によって外部記憶装置１０中の該当ファイルから読み出させたデータが暗号化されていたときは復号化したデータに対して上書きした後、一方、暗号化されていなかったときはそのままのデータに対して上書きした後、Ｉ／Ｏ手段９によって外部記憶装置１０中の該当ファイルに書き戻させるようにしている。

【００１０】これらの際に、リード手段６がファイルから読み出した暗号化されていたデータを復号化して同一サイズのデータにし、あるいはライト手段７が暗号化されていないデータを暗号化して同一サイズのデータにしてファイルに書き込むようにしている。

【００１１】また、ユーザ毎あるいはファイル毎にリード手段６による復号化あるいはライト手段７による復号化／暗号化の可あるいは不可を登録するようにしている。従って、アクセス依頼に対応してリード手段６あるいはライト手段７が暗号化されているファイルからアクセス要求のあった部分を取り出して復号化あるいは復号化してライトした後に暗号化して書き戻すことにより、暗号化されたファイルへのアクセスを高速かつ簡易にしかも部分的に行い、セキュリティレベルを高めることが可能となる。

【００１２】

【発明の実施の形態】次に、図１から図９を用いて本発明の実施の形態および動作を順次詳細に説明する。ここで、図示外の記録媒体あるいは外部記憶装置であるハードディスク装置などから読み出したプログラム、または回線を介してセンタから転送を受けたプログラムを主記憶にローディングして起動し、以下に説明する各種処理を行うようにしている。

【００１３】図１は、本発明のシステムブロック図を示す。図１において、ユーザ認証手段２は、ユーザの認証（資格検査）を行うものである（図３を用いて後述する）。

【００１４】ドライバ４は、ファイルのオープン処理、リード処理、ライト処理、クローズ処理を行うものであって、オープン手段５、リード手段６、ライト手段７、クローズ手段８などから構成されるものである。

【００１５】オープン手段５は、外部記憶装置１０中のファイルをオープンするものである（図４を用いて後述する）。リード手段６は、外部記憶装置１０中のファイルのデータを読み出させて暗号化されているときは同一サイズのデータに復号化してリード要求元に返したり、暗号化されていなかったときはそのままデータをリード要求元に返したりなどするものである（図５を用いて後述する）。

【００１６】ライト手段７は、外部記憶装置１０中のファイルのデータを読みださせて暗号化されているときは同一サイズのデータに復号化してライト要求のあったデータを上書きし、一方、暗号化されていなかったときはデータにライト要求のあったデータを上書きし、元のファイルに書き戻させたりなどするものである（図６を用いて後述する）。

【００１７】クローズ手段８は、外部記憶装置１０中のファイルをクローズするものである（図７を用いて後述する）。次に、図２のフローチャートに示す順序に従い、図１の初期処理を詳細に説明する。

【００１８】図２は、本発明の初期処理フローチャートを示す。図２において、Ｓ１は、ＯＳ（オペレーティングシステム）がドライバ確認コマンドを送信する。

【００１９】Ｓ２は、Ｓ１で送信されたドライバ確認コマンドを受信したドライバ（＃１）４が処理処理ＯＫか判別する。ＯＫのときはＯＫ応答をＯＳに返してＳ４でＯＳにドライバ（＃１）として登録する。一方、ＮＧの場合には、ＮＧ応答をＯＳに返してＳ３でエラー処理（ドライバ（＃１）がエラーとして登録などの処理）する。

【００２０】Ｓ５は、Ｓ４でドライバ（＃１）としてＯＳに登録されたので、ＯＳが初期処理の指示をドライバ（＃１）に通知する。Ｓ６は、Ｓ５の初期処理の通知に対応して、ドライバ（＃１）がファイルシステムのイベントをフックする要求を発行する。具体的には、フック時にコールされるプログラムのアドレス（例えば“ＦＳＨｏｏｋ　（）；”など）をＯＳに指定する。

【００２１】Ｓ７は、Ｓ６の要求に対応して、ＯＳがフックアドレスなどを登録した後、その応答を返す。これは、例えば後述する図８のマネージャテーブルにドライバ（＃１）に対応づけフックアドレスＦＳＨｏｏｋ（）；を登録する。

【００２２】Ｓ８は、Ｓ７に応答を受信したドライバ（＃１）がＳ６の要求についてＯＫか判別する。ＹＥＳの場合には、初期処理を完了したので終了する。ＮＯの場合には、要求失敗と判断し、再試行あるいはエラーと

して中止などする。

【００２３】以上によって、ドライバ４の初期処理が完了し、動作状態に設定されたこととなる。図３のフローチャートに示す順序に従い、図１の認証処理を詳細に説明する。

【００２４】図３は、本発明のユーザ認証フローチャートを示す。図３において、Ｓ１１は、アプリケーション１に対して、ユーザがユーザＩＤ、パスワードの入力を行う。

【００２５】Ｓ１２は、Ｓ１１で入力されたユーザＩＤ、パスワードをもとにテーブルを参照してチェックを行い、ＯＫか判別する。ＹＥＳの場合には、Ｓ１３に進む。ＮＯの場合には、エラーとしてＳ１１で再入力あるいはエラーとして中止する。

【００２６】Ｓ１３は、ユーザまたはアプリケーションからの暗号化／復号化フォルダ（ディレクトリ）の指定を行う。Ｓ１４は、ドライバへの暗号化／復号化フォルダの指定を行う。

【００２７】Ｓ１５は、Ｓ１４で通知を受けたドライバ（＃１）が暗号化／復号化フォルダの設定・保存を行う。これは、例えば後述する図９に示すように、ユーザ毎に、フォルダ名に対応づけてリード処理／ライト処理毎に暗号化／復号化するものを登録する（○は暗号化／復号化する、×は暗号化／復号化しない）。

【００２８】Ｓ１６は、暗号化／復号化を開始指示する。Ｓ１７は、ドライバ（＃１）がＳ１６の指示に対応して暗号化／復号化を開始する。

【００２９】以上によって、認証されたユーザ毎にフォルダ毎に図９に示すようにリード処理およびライト処理について暗号化／復号化するか否かを登録し、当該登録に従い暗号化／復号化を開始する。

【００３０】図４のフローチャートに示す順序に従い、図１のオープン処理を詳細に説明する。図４は、本発明のオープン処理フローチャートを示す。

【００３１】図４において、Ｓ２１は、アプリケーション１がオープン関数を発行する。Ｓ２２は、ファイルシステム３がオープン処理を行う。Ｓ２３は、ドライバ（＃１）４が図示の下記のオープンイベントをＯＳより受信する（これは、ドライバの登録時に、ＯＳに通知してもらうイベント、通知時にコールされる暗号化／復号化プログラムのアドレスを登録することで実現されている）。

【００３２】・ファイル名（パス名を含む）：
・ファイル属性：
・ファイルハンドル：
・オープン関数のパラメタ：
・ファイルシステムへの復帰アドレス：
・その他

Ｓ２４は、フォルダが暗号化／復号化対象かのチェックがＯＫか判別する。フォルダ名はパス名（ファイル名）より抽出する。ＹＥＳの場合（対象のファイルが存在するなどの場合）には、Ｓ２７に進む。ＮＯの場合には、暗号化／復号化対象外ファイルとしてＳ２５で復帰アドレスへ復帰し、オープン処理に戻る。

【００３３】Ｓ２７は、ファイル属性から処理しないファイルを振り分ける。これは、図示の下記のファイル属性から処理する必要のないファイルを除外する。

・システムファイル：
・隠しファイル：
・ドライバファイル：
・その他

ＹＥＳの場合には、暗号化／復号化の対象外のファイルと判明したので、Ｓ２８で属性テーブルにファイルハンドルを保存し、Ｓ２９に進む。一方、Ｓ２７のＮＯの場合には、Ｓ２９に進む。

【００３４】Ｓ２９は、復帰アドレスへ復帰する。Ｓ３０は、オープン処理を完了する。Ｓ３１は、オープン関数に復帰する。

【００３５】以上の手順によって、アプリケーション１がオープン関数を発行し、ファイルのオープン処理を完了したこととなる。そして、後述する図５のリード処理あるいは図６のライト処理に進む。

【００３６】図５のフローチャートに示す順序に従い、図１のリード処理を詳細に説明する。図５は、本発明のリード処理フローチャートを示す。

【００３７】図５において、Ｓ４１は、アプリケーション１がリード関数を発行する。Ｓ４２は、ファイルシステム３がリード処理を行う。Ｓ４３は、ドライバ（＃１）４が図示の下記のリードイベントを受信する。

【００３８】・ファイル名（パス名を含む）：
・ファイルハンドル：
・リードポイント：
・リードバッファ：
・リードサイズ：
・ファイルシステムへの復帰アドレス：
・その他

Ｓ４４は、フォルダが暗号化／復号化対象かチェックがＯＫか判別する。ＹＥＳの場合（暗号化／復号化対象の場合）には、Ｓ４６に進む。ＮＯの場合には、通常の処理としてＳ４５で復帰アドレス（ＯＳ）へ復帰し、通常のリード処理を行う。

【００３９】Ｓ４６は、オープン処理時に更新した属性テーブルに当該要求のファイルハンドルがあるか否かをチェックする。これは、ファイルハンドルから暗号化／復号化の対象外のファイルか判別する。ＮＯの場合には、Ｓ４８に進む。ＹＥＳの場合には、Ｓ４７で復帰アドレスへ復帰し、通常のリード処理をする。

【００４０】Ｓ４８は、リード処理を呼び出す。これは、外部記憶装置１０である例えばＨＤＤ（ハードディスク装置）のドライバを呼び出し、バウンダリ処理を行

ってデータをリードする。
【００４１】Ｓ４９は、Ｓ４８で呼び出された例えばＨＤＤドライバがＨＤＤからバウンダリを考慮してデータを読み出す。Ｓ５０は、Ｓ４９で読み出したデータについて、同一サイズで復号化処理を行う。ここで、８バイト単位での暗号化処理、および端数処理を行う。これにより、８バイト単位に復号化処理を行い、読み出したデータの８バイトと、復号化した後のデータが８バイトとなるようにしており、復号化前と復号化後のデータが共に８バイトで同一とし、データの長さから両者が区別つかないようにし、セキュリティレベルを高めている。
【００４２】Ｓ５１は、リードサイズ分をリードバッファに設定する。Ｓ５２は、復帰アドレスへ復帰する。Ｓ５３は、リード処理を完了する。
【００４３】Ｓ５４は、リード関数に復帰する。以上によって、アプリケーション１がリード関数を発行すると、発行されたリード関数で指定されたデータがここでは、８バイト単位にファイルから読みだされて暗号化されているときは復号化し、暗号化されていないときは図示しないがそのままでリードバッファに設定することをリード分のデータ分だけ繰り返し、アプリケーション１はリードバッファから読み出して一連のリード処理を行うことが可能となる。
【００４４】図６のフローチャートに示す順序に従い、図１のライト処理を詳細に説明する。図６は、本発明のライト処理フローチャートを示す。
【００４５】図６において、Ｓ６１は、アプリケーション１がライト関数を発行する。Ｓ６２は、ファイルシステム３がライト処理を行う。Ｓ６３は、ドライバ（＃１）４が図示の下記のライトイベントを受信する。
【００４６】・ファイル名（パス名を含む）：
・ファイルハンドル：
・ライトポイント：
・ライトバッファ：
・ライトサイズ：
・ファイルシステムへの復帰アドレス：
・その他
Ｓ６４は、フォルダが暗号化／復号化対象かのチェックがＯＫか判別する。ＹＥＳの場合（暗号化／復号化対象の場合）には、Ｓ６６に進む。ＮＯの場合には、Ｓ６５で復帰アドレスへ復帰し、通常のライト処理をする。
【００４７】Ｓ６６は、オープン処理時に更新した属性テーブルに当該要求のファイルハンドルがあるか否かをチェックする。これは、ファイルハンドルから暗号化／復号化対象外のファイルか判別する。ＮＯの場合には、Ｓ６８に進む。ＹＥＳの場合には、Ｓ６７で復帰アドレスへ復帰し、通常のライト処理をする。
【００４８】Ｓ６８は、リード処理を呼び出す。これは、外部記憶装置１０である例えばＨＤＤ（ハードディスク装置）のドライバを呼び出し、バウンダリ処理を行ってデータをリードする。
【００４９】Ｓ６９は、Ｓ６８で呼び出された例えばＨＤＤドライバがＨＤＤからバウンダリを考慮してデータを読み出す。Ｓ７０は、Ｓ６９で読み出したデータについて、同一サイズで復号化処理を行う。ここで、８バイト単位での復号化処理、および端数処理を行う。これにより、８バイト単位に復号化処理を行い、読み出したデータの８バイトと、復号化した後のデータが８バイトとなるようにしており、復号化前と復号化後のデータが共にに８バイトで同一とし、データの長さから両者が区別つかないようにし、セキュリティレベルを高めている。
【００５０】Ｓ７１は、ライトデータを復号化データに上書きする。Ｓ７２は、Ｓ７１で上書きしたデータについて、同一サイズで暗号化処理を行う。ここで、８バイト単位での暗号化処理、および端数処理を行う。これにより、８バイト単位に暗号化処理を行い、上書きしたデータの８バイトと、暗号化した後のデータが８バイトとなるようにしており、暗号化前と暗号化後のデータが共にに８バイトで同一とし、データの長さから両者が区別つかないようにし、セキュリティレベルを高めている。
【００５１】Ｓ７３は、ライト処理を呼び出す。これは、外部記憶装置１０であるＨＤＤのＨＤＤドライバを呼び出し、バウンダリ処理を行ってデータをライトする。Ｓ７４は、Ｓ７３で呼び出されたＨＤＤドライバがＨＤＤからバウンダリを考慮してデータを書き戻す。
【００５２】Ｓ７５は、ライト処理を完了する。Ｓ７６は、ドライバから復帰する。Ｓ７７は、ライト関数に復帰する。
【００５３】以上によって、アプリケーション１がライト関数を発行すると、発行されたライト関数で指定されたデータがここでは、８バイト単位にファイルから読みだされて暗号化されているときは復号化し、暗号化されていないときは図示しないがそのままでとし、これらデータに対してライトデータを上書きした後、暗号化されていたときは暗号化したデータを、暗号化されていなかったときはそのままのデータを８バイト単位にファイルに書き戻すことを繰り返して一連のライト処理を行うことが可能となる。
【００５４】図７のフローチャートに示す順序に従い、図１のクローズ処理を詳細に説明する。図７は、本発明のクローズ処理フローチャートを示す。
【００５５】図７において、Ｓ８１は、アプリケーション１がクローズ関数を発行する。Ｓ８２は、ファイルシステム３がクローズ処理を行う。Ｓ８３は、ドライバ（＃１）４が図示の下記のクローズイベントを受信する。
【００５６】・ファイル名：
・ファイル属性：
・ファイルシステムへの復帰アドレス：
・その他

Ｓ８４は、フォルダのチェックがＯＫか判別する。ＹＥＳの場合（クローズ対象のファイルが存在するなどの場合）には、Ｓ８６に進む。ＮＯの場合には、Ｓ８５で復帰アドレスへ復帰し、通常のクローズ処理をする。

【００５７】Ｓ８６は、オープン処理時に更新した属性テーブルに当該要求のファイルハンドルがあるか否かをチェックする。ＹＥＳの場合には、Ｓ８９に進む。ＮＯの場合には、Ｓ８７で復帰アドレスへ復帰し、通常のクローズ処理をする。

【００５８】・システムファイル：
・隠しファイル：
・ドライバファイル：
・その他
Ｓ８９は、属性テーブルに保存していたファイルハンドルを消去し、Ｓ９０に進む。

【００５９】Ｓ９０は、復帰アドレスへ復帰する。Ｓ９１は、クローズ処理を完了する。Ｓ９２は、クローズ関数に復帰する。

【００６０】以上の手順によって、アプリケーション１がクローズ関数を発行し、ファイルのクローズ処理を完了したこととなる。図８は、本発明のマネージャテーブル例を示す。これは、既述した図２の初期処理中のＳ７でＯＳに登録されたものであって、ここでは、図示の下記のようにドライバ毎にフックアドレスを登録し、データの授受を実現するためのものである。

【００６１】ドライバ名　　フックアドレス
ドライバ＃１　　フックアドレス　ＦＳＨｏｏｋ（）；
図９は、本発明のユーザレベル管理テーブル例を示す。これは、ユーザ毎にフォルダ毎にリード処理およびライト処理の可否を登録するものであって、既述した図３のＳ１５などで登録されるものである。このユーザレベル管理テーブルにより、ユーザ毎かつフォルダ毎に既述したリード処理およびライト処理の可否を参照し、既述した図５のリード処理、図６のライト処理を実行することにより、セキュリティを確保することが可能となる。

【００６２】
【発明の効果】以上説明したように、本発明によれば、アクセス依頼に対応してリード手段６あるいはライト手段７が暗号化されているファイルからアクセス要求のあった部分を取り出して復号化あるいは復号化してライトした後に暗号化して書き戻す構成を採用しているため、暗号化されたファイルへのアクセスを高速かつ簡易にしかも部分的に行うことができる。これらにより、

（１）　本願発明は、暗号化前と暗号化後、復号化前と復号化後のデータの長さを同一（例えば８バイトで同一）としているため、ファイル内のデータが暗号化されているか否かをデータ長（ファイルサイズ）から判別できず、セキュリティレベルを高めることができる。

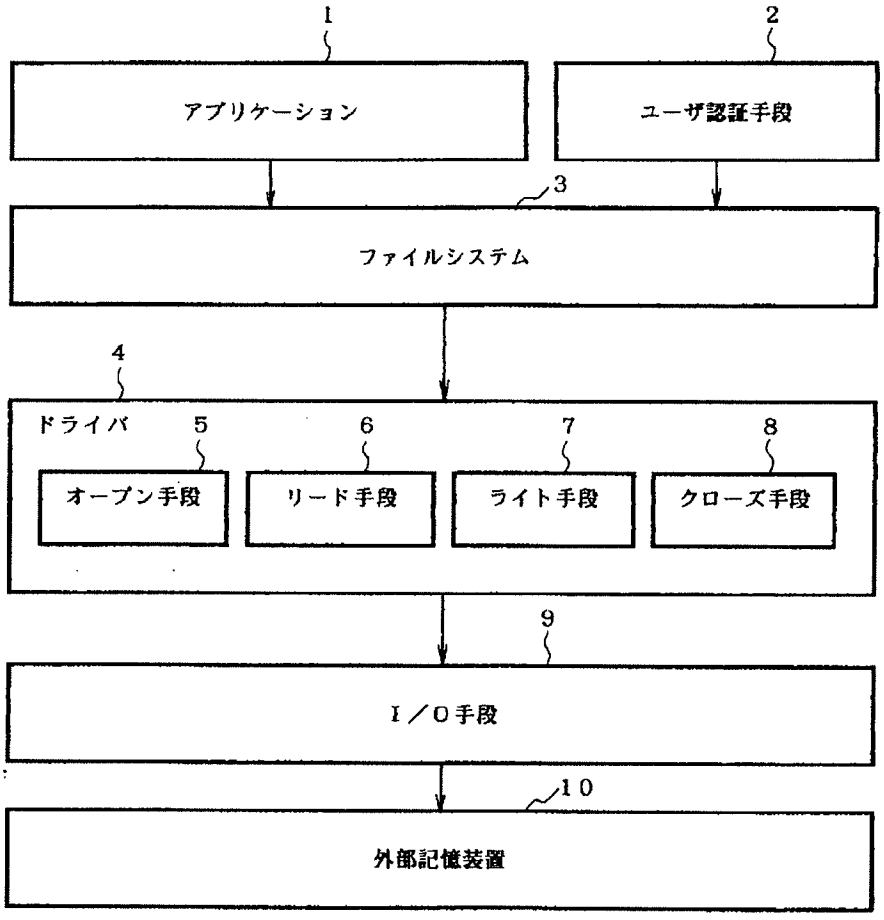
【００６３】（２）　本願発明は、部分的（例えば８バイト単位）にファイルから暗号化データを読み出して復号化したり、部分的（例えば８バイト単位）にファイルから暗号化データを読み出して復号化しライトデータを上書きした後に暗号化してファイルに書き戻しているため、リード処理／ライト処理で必要な部分データのみをアクセスし、高速、迅速かつセキュリティレベルを高くできる。

【００６４】（３）　従来のファイルから全データを作業ファイルに読み出して復号化したデータを、アプリからアクセスするような場合には復号化したデータが作業ファイルに存在してデータの内容が判明してしまいセキュリティレベルが低い欠点があるが、本願発明では作業ファイルがなく、リード処理あるいはライト処理毎にその都度、ファイルからデータを部分的（例えば８バイト単位）に読み出して復号化したり、データを部分的（例えば８バイト単位）に読み出して復号化して上書きした後に暗号化してファイルに書き戻してしまい、セキュリティレベルが非常に高い。

【図面の簡単な説明】
【図１】本発明のシステムブロック図である。
【図２】本発明の初期処理フローチャートである。
【図３】本発明のユーザ認証フローチャートである。
【図４】本発明のオープン処理フローチャートである。
【図５】本発明のリード処理フローチャートである。
【図６】本発明のライト処理フローチャートである。
【図７】本発明のクローズ処理フローチャートである。
【図８】本発明のマネージャテーブル例である。
【図９】本発明のユーザレベル管理テーブル例である。
【符号の説明】
１：アプリケーション
２：ユーザ認証手段
３：ファイルシステム
４：ドライバ
５：オープン手段
６：リード手段
７：ライト手段
８：クローズ手段
９：Ｉ／Ｏ手段
１０：外部記憶装置

【図1】

本発明のシステムブロック図



【図8】

本発明のマネージャテーブル例

ドライバ#1	フックアドレス　FSHook( );
ドライバ#	・
・	・

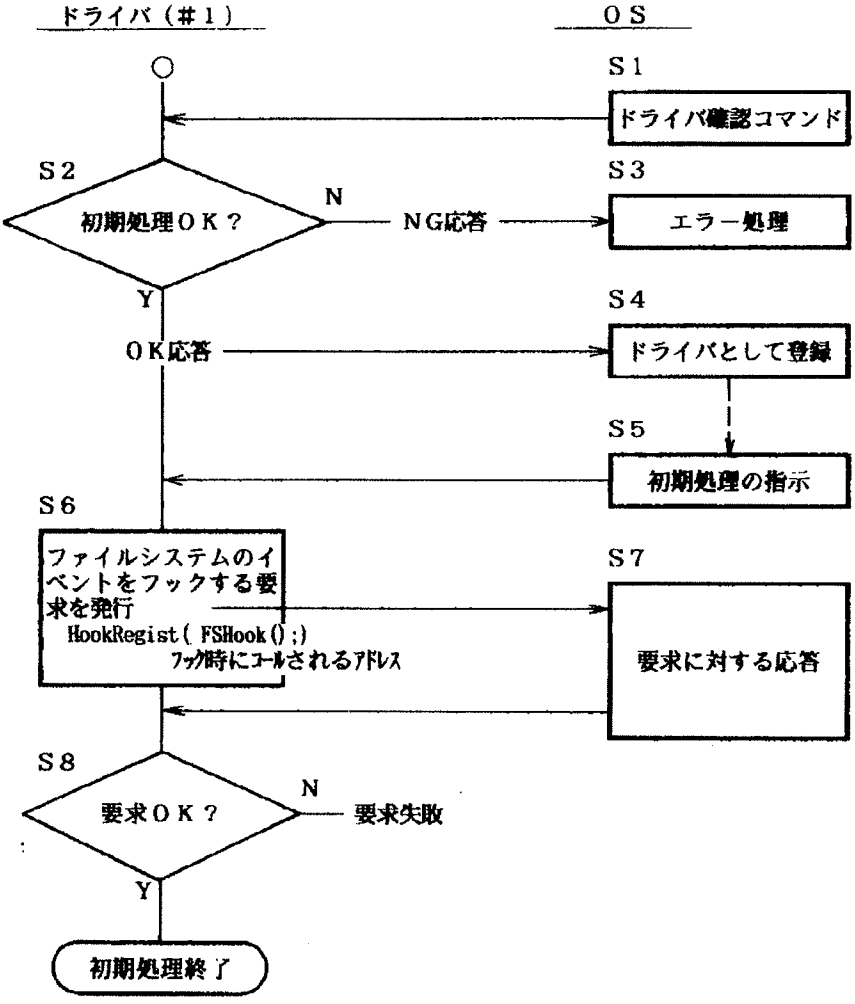
【図9】

本発明のユーザレベル管理テーブル例

フォルダ名	リード処理	ライト処理
フォルダA	○	○
フォルダB	○	×
フォルダC	×	○
フォルダD	×	×

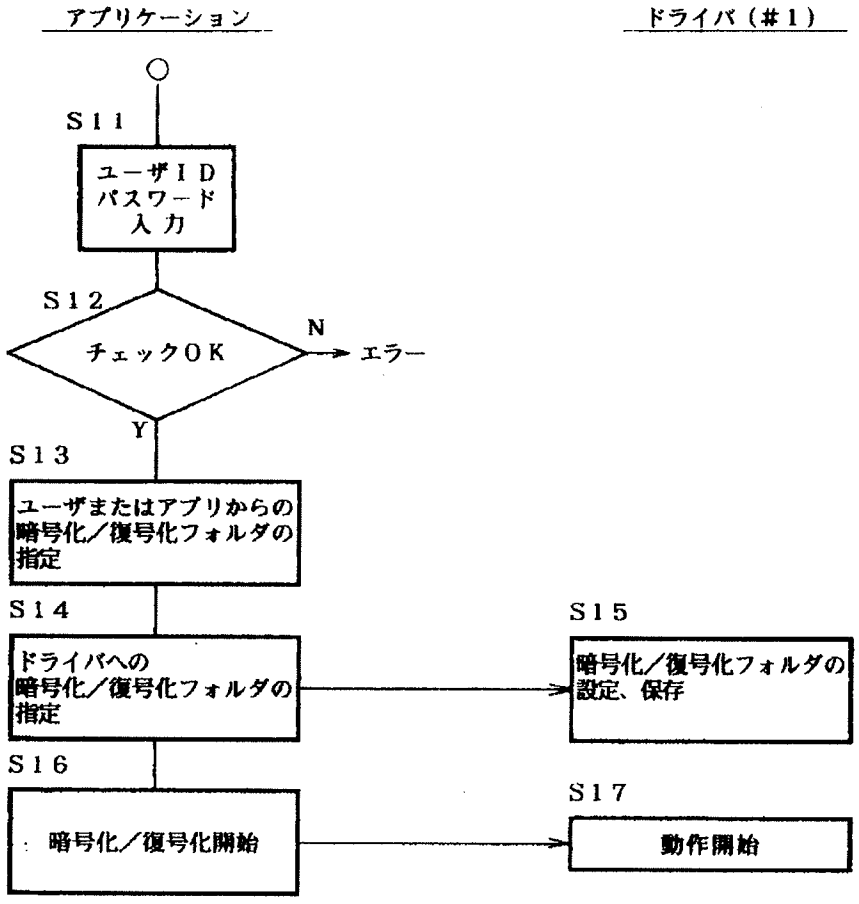
【図2】

本発明の初期処理フローチャート



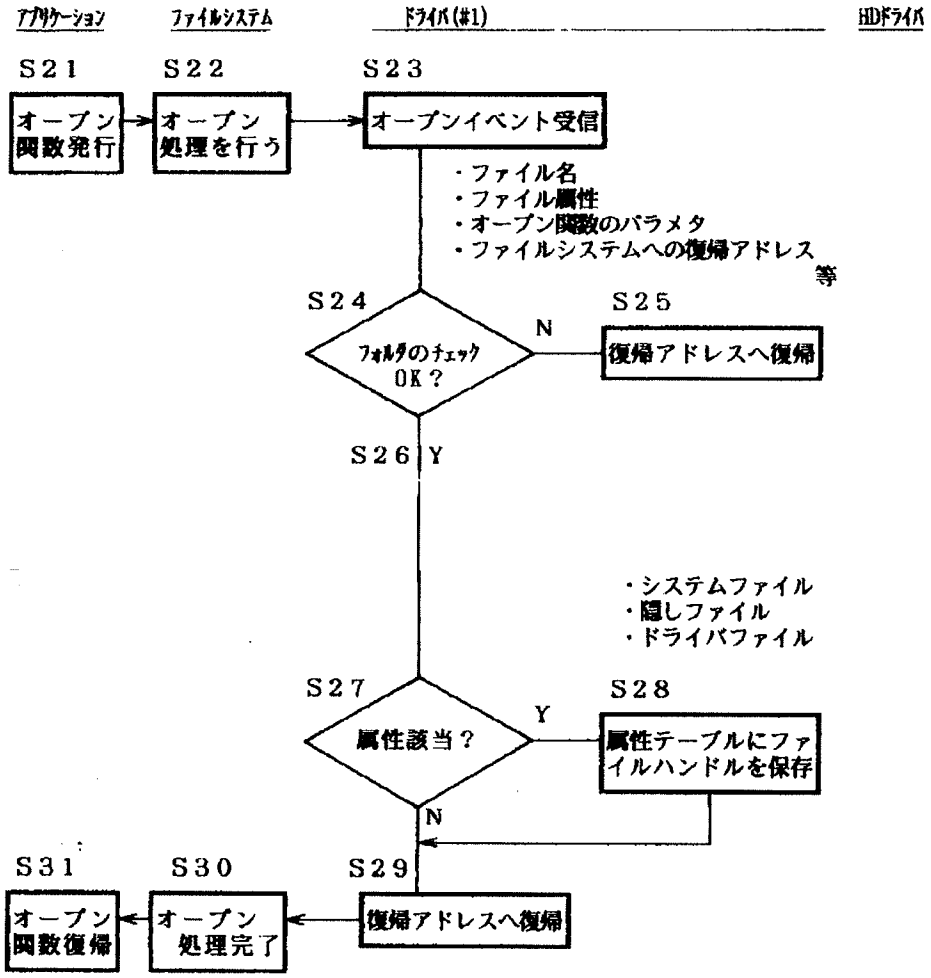
【図3】

本発明のユーザ認証フローチャート



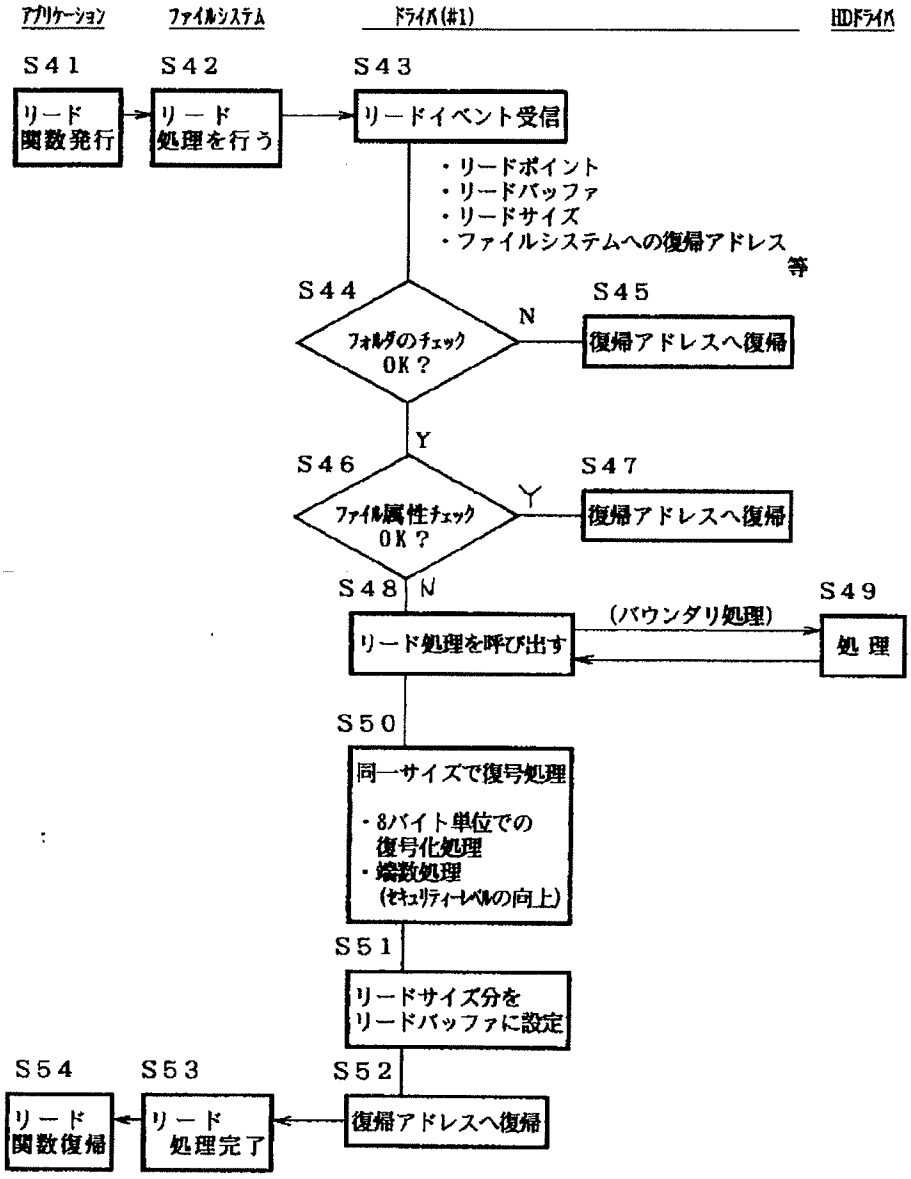
【図4】

本発明のオープン処理フローチャート



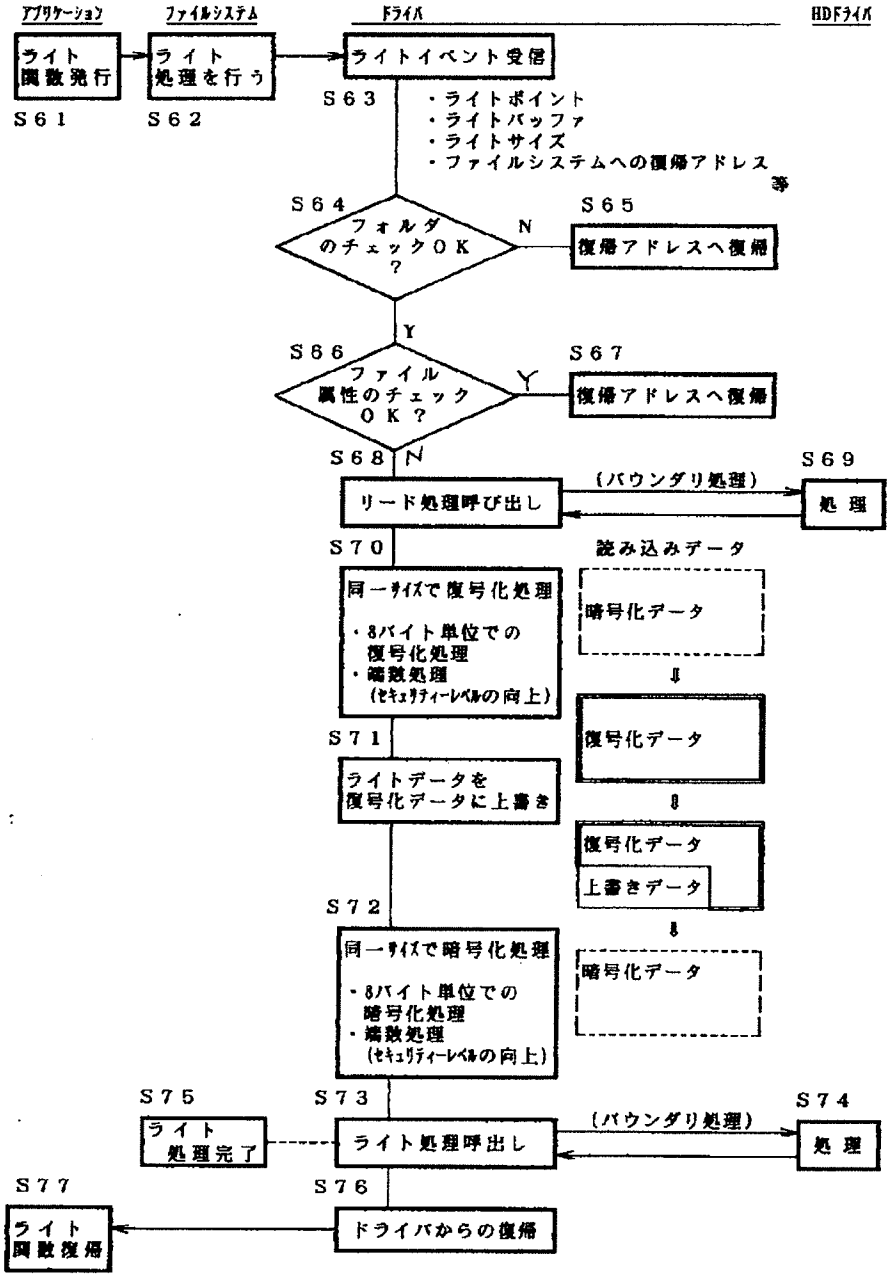
【図5】

本発明のリード処理フローチャート



【図6】

本発明のライト処理フローチャート



【図7】

本発明のクローズ処理フローチャート